The Software Management Environment (SME)

Jon D. Valett
(NASA/GSFC)
William Decker
and
John Buell
(Computer Sciences Corporation)

1.0. Background (charts 1 and 2)

The Software Management Environment (SME) is a research effort designed to utilize the past experiences and results of the Software Engineering Laboratory (SEL) [Card82] and to incorporate this knowledge into a tool for managing projects. SME provides the software development manager with the ability to observe, compare, predict, analyze, and control key software development parameters such as effort, reliability, and resource utilization. This paper describes the major components of the SME, outlines the architecture of the system, and provides examples of the functionality of the tool.

The SEL has been researching and evaluating software development methodologies for over ten years. This research has provided valuable insight into the software development process of one particular organization. By collecting detailed software development data and recording that data in a software engineering data base [Church82][Heller87], the SEL has been able to characterize and understand the development process within that organization. Using this data to measure the impact of various methodologies, tools, and perturbations to that process has enabled the SEL to better control and manage the software projects of this organization.

Recognizing the vast potential of providing the experience of previous projects, the data, the research results, and the knowledge of experienced software managers to the managers of ongoing projects, research efforts were initiated to provide these items in the form of a tool. Initial prototype efforts began in 1984, with the development of a tool that explored the possibilities of providing this information. That effort was thoroughly analyzed and requirements were developed for a more complete software system late in 1986 [Valett87]. During this time work began on the current SME.

The major functionality that the SME provides for its user can be divided into four high level concepts:

1.) The ability for a manager to compare the ongoing software project to other projects. This function allows the manager to view software metric data such as weekly effort or error data and to compare it to other projects.

2.) The ability for the manager to receive predictions of future events of interest. SME will predict the final values for key project parameters such as effort or reliability.

3.) SME will also analyze project data to give insights into the strengths and weaknesses of the development process.

4.) SME will analyze overall project quality. This will

provide the manager with high-level insight into the project's overall development process.

Thus, the SME enables the manager to gain valuable insight into the progress and quality of a software development project.

This paper describes the concepts and architecture of the SME. Section 2.0 is devoted to describing the research results and data which are incorporated into the SME. Section 3.0 describes the architecture of the system and gives examples of the functions available to the manager. Finally, a brief discussion is presented in section 4.0.

2.0 The Components of SME

Attempting to integrate past research results along with dynamic project data, the SME provides the manager with a wide variety of information for monitoring and controlling an ongoing software project. The information required to provide this functionality can be broken into three major components: 1) the corporate history, 2) research results from studies of the software development process, and 3) management rules for software development.

2.1 The Corporate History (charts 3 and 4)

One underlying assumption of the SME is that a corporate history of some type exists. In this case, the SEL data base serves as the corporate memory for the SME. The SEL data base has evolved into its current form over the nearly 12 years of its existence. The data base itself provides the SME with the majority of the raw data required to monitor a project.

The major items of data provided by the data base include weekly software parameters that are of interest to the software manager. These weekly items of data include such parameters as effort, computer utilization, growth of source code, change history, and error history. All of these items are available as part of the SEL data base for any project of interest, as well as on the past projects that a manager may want to use as a basis for comparison.

Many of the other data needed by the SME is acquired from the SEL data base. This data includes items which characterize the types of projects as well as the language or tools used. Subjective data which is used to evaluate projects on a series of software methodology questions is also used by the system.

During the 12 years of the SEL's existence, numerous studies and reports characterizing and evaluating the software development environment have been written. These studies and reports have provided numerous research results for the environment. Thus, the SEL data base establishes the foundation for all of the components of the SME.

2.2 Research Results (chart 5)

A second major component of the SME is the research results that have been developed via the SEL data base. Information

derived from papers and studies developed through experimentation and through analysis of the SEL data base is a key part of the SME (for examples of results see [Valett88]). The SME attempts to incorporate these research results via models and measures for the software environment. Based on a comprehensive understanding of the development environment, these models and measures are used by the SME to enable the manager to better understand how a particular project compares to the normal project within the environment. They also are used by the SME in predicting and estimating future conditions on the software project.

Models of software development parameters are essential for the SME to perform its prediction and comparison functions. A model profiles the expenditure, the utilization, or the production of a software development parameter. As an example, a model of the staffing profile would capture the typical expenditure of effort over the entire software development life cycle [Basili78]. This type of model can be used by a manager to compare the current effort expenditure with the typical one for this environment.

Other types of relationships are used by the SME to capture known affects of specific software development methodologies. For instance, the knowledge that code reading is the most affective method for finding errors in this environment [Selby87], is important information to disseminate to a manager. One goal of the SME is to provide a knowledge base of known facts and relationships about a particular environment.

2.3 Software Development Rules (chart 6)

A final major component of the SME is software development rules. The SME attempts to integrate the experience of software managers into an expert system concept to provide the ability to analyze project measures and status. Previously, this experience was only captured in lessons learned or summary documents. The SME formalizes this knowledge into a basic structure that will continually evolve as the experience and knowledge are validated. By automating the knowledge utilization into an expert system, SME gives the manager the ability to apply past experience to current projects. The basic concept of utilizing expert systems for software management was proven feasible by previous research done by the SEL [Valett85][Ramsey86]. Admittedly, the extension of these concepts for use within the SME is an extremely difficult area of research, however, early results show they will be very useful.

Within the SME experienced manager's knowledge can be used in numerous areas. The knowledge has been collected from interviews with numerous managers, along with analysis of SEL data and information obtainable from the various reports and studies written by the SEL. An example of the type of knowledge used by the SME is shown in chart 6. This rule:

If error rate is lower than normal then
1. Insufficient testing
2. Experienced team
3. Problem less difficult than expected

is a simplified form of the type of rule collected for use in the

SME. Utilizing this rule, numerous other rules, and facts about the measures and status of the software project, the SME can reach conclusions pertaining to the deviations of project measures, such as error rate. Thus, the system can give the manager vital information regarding the strengths and weaknesses of a software development effort. In the future, this knowledge will also be used to provide the overall assessment functionality of the SME.

Obviously, the collection and validation of these rules and relationships is a major task. The research into this part of the SME will involve continual iteration and evolution. However, by establishing a baseline set of software rules and incorporating them into the SME and by constantly integrating feedback on the validity of the conclusions and knowledge, the SME knowledge base will mature into an even more valuable component of the system.

3.0 SME Architecture and Functionality

The SME architecture is designed to integrate the three major components described in section 2 into a tool which provides the manager with the functions of comparison, analysis, prediction, and expert guidance (see chart 7). The major processing of the system is performed on a VAX 11/780 and is written in Pascal, with the user interface and some data handling procedures performed on IBM/PC compatibles. The selection of this particular hardware architecture was driven by the desire to make SME accessible to managers in their offices and to provide color graphics capabilities. The remainder of this section is devoted to describing the major functionality of the SME: comparison, analysis, and prediction.

3.1 Comparison (charts 8 and 9)

The comparison function of the SME is designed to allow the manager to view project data on measures of interest such as effort, lines of code (LOC), CPU utilization, etc. and to compare these measures to past projects and to models of the normal project. Comparison utilizes the SEL data base and current project data along with models and measures of the typical project. Providing the comparison feature allows the manager to determine how the current project is behaving as it compares to past similar projects as well as whether or not the current project is following the "typical" pattern for that particular measure. In the examples chart 8 shows a comparison of the number of errors on a current project against the number errors on a past project, while chart 9 shows a similar comparison, except that the past project is replaced by a model of errors committed for the environment. These types of comparisons are available for a variety of project measures; they enable the manager to examine the characteristics of the current project in the context of other projects.

3.2 Analysis (chart 10)

Giving the user the knowledge of experienced software managers, the analysis function provides insights into the strengths and weaknesses of a project. Utilizing the SEL database, the current data, the models and measures, and the rule base, the analysis function compares the value for a certain measure for a current project to the model of that measure and reaches conclusions about why the project is deviating from the norm. The example shows a comparison of the number of errors on the current project with the model for errors. Since the number of errors is below what would be expected at this point in the software development, the SME can provide analysis as to why this condition may be occurring. The example illustrates a use of the rule discussed in section 2.3. While this is an elementary example, it does show the type of information SME provides. This type of analysis provides the manager with valuable insight into potential problems that might be occurring on the project of interest.

3.3 Prediction (chart 11)

Based on the current status of a software measure, the prediction function attempts to estimate the behavior of the measure through the completion of the project. Making heavy use of the models and measures along with the data for the project of interest, this function gives managers reasonable estimates of key project parameters. For example, given the current system size in LOC, information regarding the project's subjective profile, and some project estimates, SME predicts the final system size. Similarly, information on the current phase and error rate of a project along with certain models and measures, enables the SME to predict the final error rate for the system. Obviously, these and other key project parameters are invaluable to the manager in planning and controlling a software project.

4.0 Discussion (chart 12)

While the SME currently provides parts of all the capabilities described in section 3, it is still considered a research effort. Much research into each of the functions described as well as into other more advanced features of the system is still required for the system to become a fully useful tool. Thus, the system will change as these features are integrated into the overall architecture of the system.

In a similar manner, the system will continually evolve as the knowledge of the environment evolves. For example, although the current SME focuses on the waterfall life cycle model, as other paradigms are utilized and adopted within the environment, these results will be factored into the SME. The SME will continue to mature as long as research into the understanding of the development environment continues to provide an improved understanding of the software process.

Continuing to focus on utilizing the knowledge and experience of past research in addition to future research, the SME provides and will continue to provide a valuable feedback mechanism which encourages the reuse of this knowledge and

experience. The formalization of this reuse into a constantly maturing software tool, ensures that the knowledge will be captured and used on future software development efforts. Thus, the SME should continue to be a useful software management tool that will provide the software development manager with valuable information and insight into the quality of a software development project.

REFERENCES

[Basili78] Basili, V. and M. Zelkowitz, "Measuring Software Development Characteristics in the Local Environment," Computers and Structures, August 1978, Vol. 10.

[Card82] Card, D., F. McGarry, G. Page, et al., The Software Engineering Laboratory, SEL-81-104, February 1982.

[Church82] Church, V., D. Card, and F. McGarry, Guide to Data Collection, SEL-81-101, August 1982.

[Heller87] Heller, G.,Data Collection Procedures for the Rehosted SEL Data Base, SEL-87-008, October 1987.

[Ramsey86] Ramsey, C. and V. Basili, "An Evaluation of Expert Systems for Software Engineering Management," TR-1708, University of Maryland, Technical Report, September 1986.

[Selby87] Selby, R. and V. Basili, "Comparing the Effectiveness of Software Testing Strategies," IEEE Transactions on Software Engineering, December 1987.

[Valett85] Valett, J. and A. Raskin, "DEASEL: An Expert System for Software Engineering," Proceedings of the Tenth Annual Software Engineering Workshop, SEL-85-006, December 1985.

[Valett87] Valett, J., "The Dynamic Management Information Tool (DYNAMITE): Analysis of Prototype, Requirements, and Operational Scenarios," Masters Thesis, The University of Maryland, May 1987.

[Valett88] Valett, J. and F. McGarry, "A Summary of Software Measurement Experiences in the Software Engineering Laboratory," Proceedings of the 21st Annual Hawaii International Conferences on System Sciences, January 1988.

THE VIEWGRAPH MATERIALS

FOR THE

J. VALETT PRESENTATION FOLLOW

THE SOFTWARE MANAGEMENT ENVIRONMENT (SME)

JON VALETT
(NASA/GSFC)

BILL DECKER
JOHN BUELL
(CSC)

SME CONCEPT

GOAL: To produce a tool (or set of tools) that utilizes historical information pertinent to software development in order to provide guidance and information for new development efforts

DRIVEN BY:

- Availability of
  - Archived data in the SEL
  - Results of SEL studies
  - Accumulation of expertise on the software development process
- Need for
  - Management guidance
  - Information availability
- Drive for research into improving the development process

SME GOALS

Provide the Software Development Manager with insights on an ongoing project via:

1. Comparison	• Compare development profile of current project with past projects
2. Prediction	• Predict future events (Cost/reliability...)
3. Analysis	• Determine strengths and weaknesses of project
4. Expert Guidance	• Determine overall quality of project

J. Valett
NASA/GSFC

C218.003

SEL PRODUCTION ENVIRONMENT

TYPE OF SOFTWARE: Scientific, ground based, interactive graphic
moderate reliability and response requirements

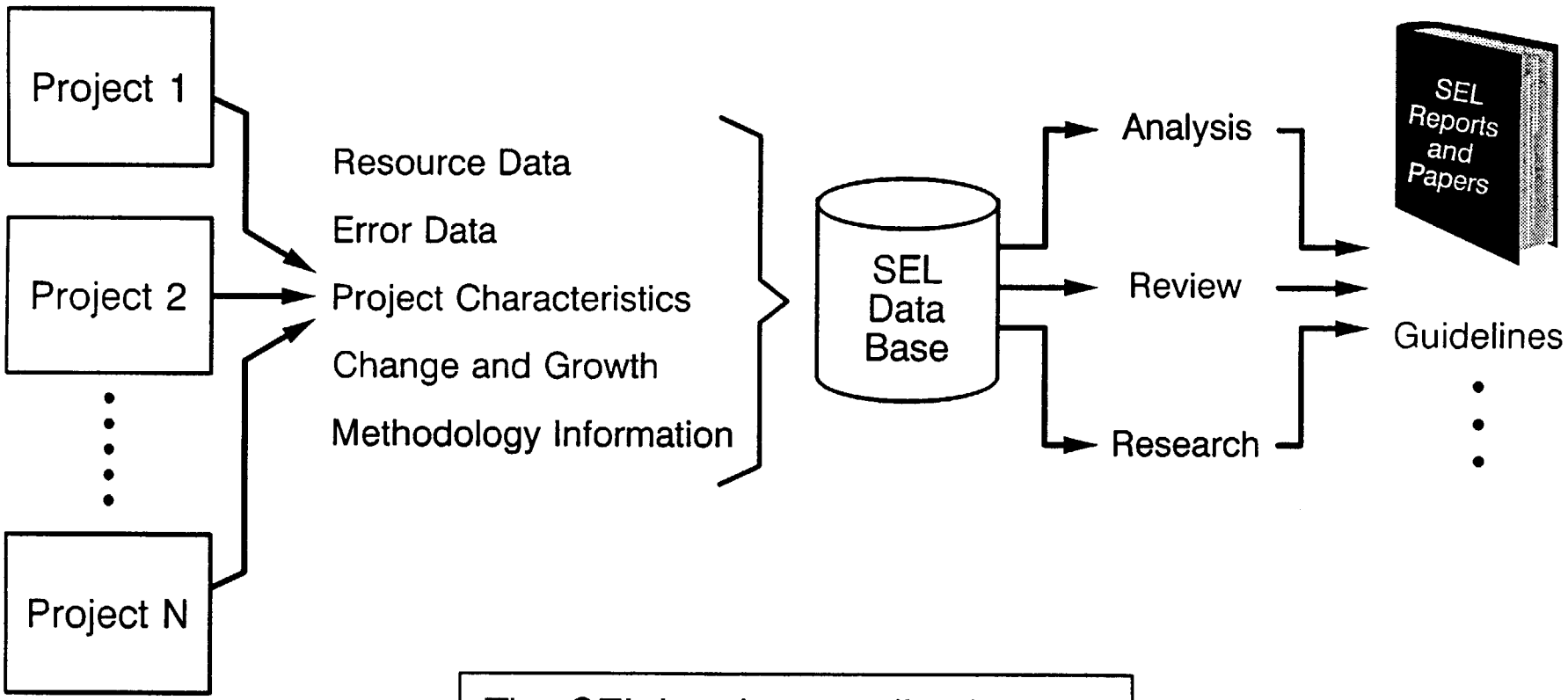
LANGUAGES: Primarily FORTRAN; some Ada and assembler

PROJECTS:
- Average duration 1-2 years (≈8 staff years)
- Average size 60-70 KSLOC (some projects 180-225 KSLOC)

EXPERIMENTATION:
- Monitor production level projects and collect data
- Apply methodologies and evaluate
- Apply results to future projects

In 1977, the SEL began collecting development data and studying the software development process. Since then over 65 projects have been monitored

SEL Collects Software Development Data

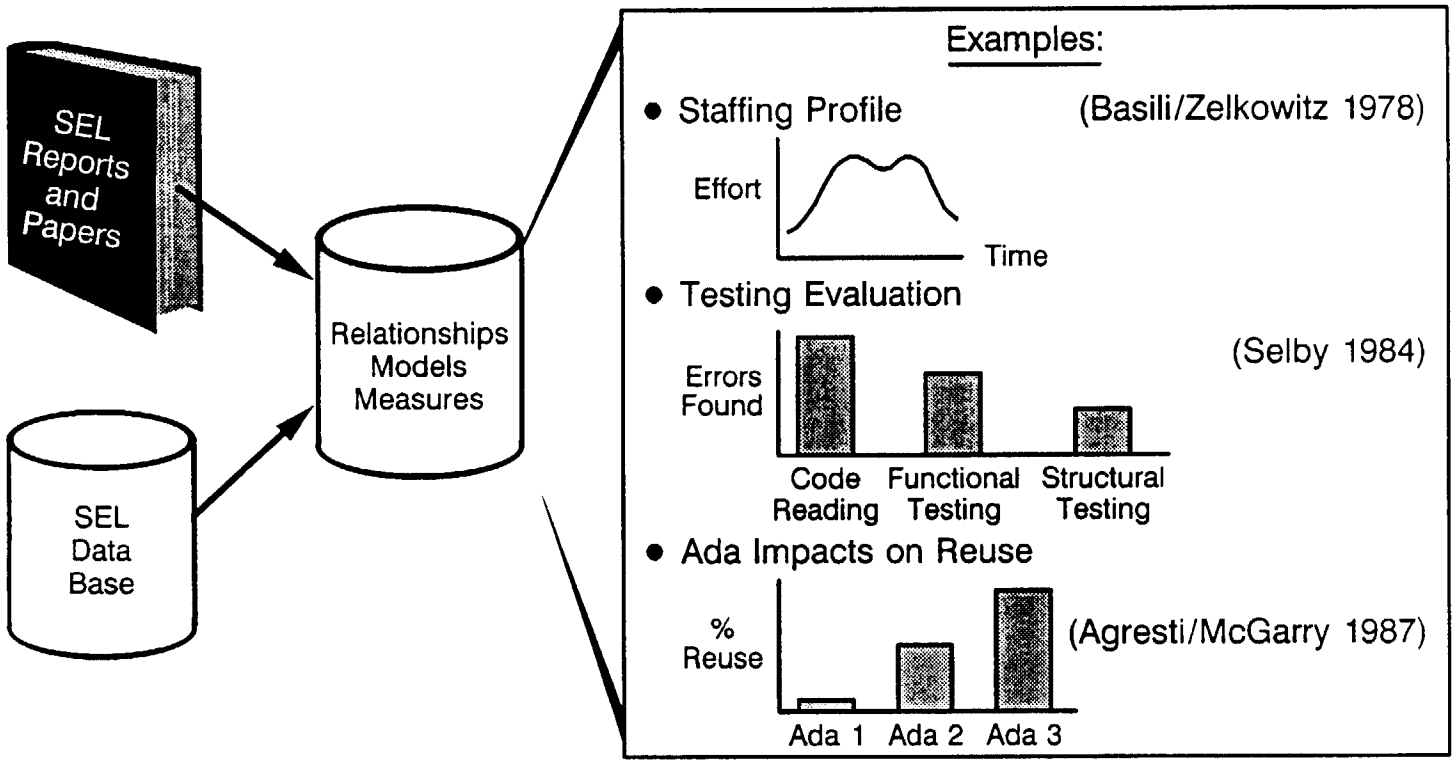


The SEL has been collecting and analyzing data for over 10 years.

J. Valett
NASA/GSFC

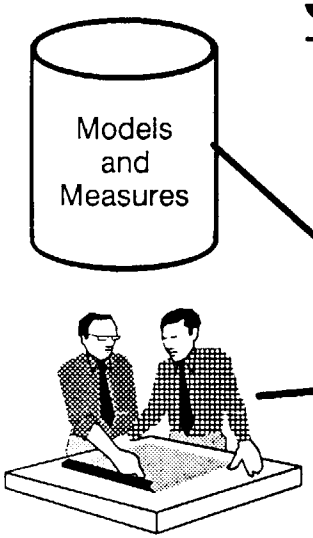
C218.005

SEL RESEARCH RESULTS



Studies of SEL projects and applied research have resulted in over 100 reports covering topics ranging from development profiles to "rules" of software development

SEL DEVELOPMENT OF SOFTWARE DEVELOPMENT RULES



Example:

If error rate is * lower than normal then;

1. Insufficient testing
2. Experienced Team
3. Problem less difficult than expected

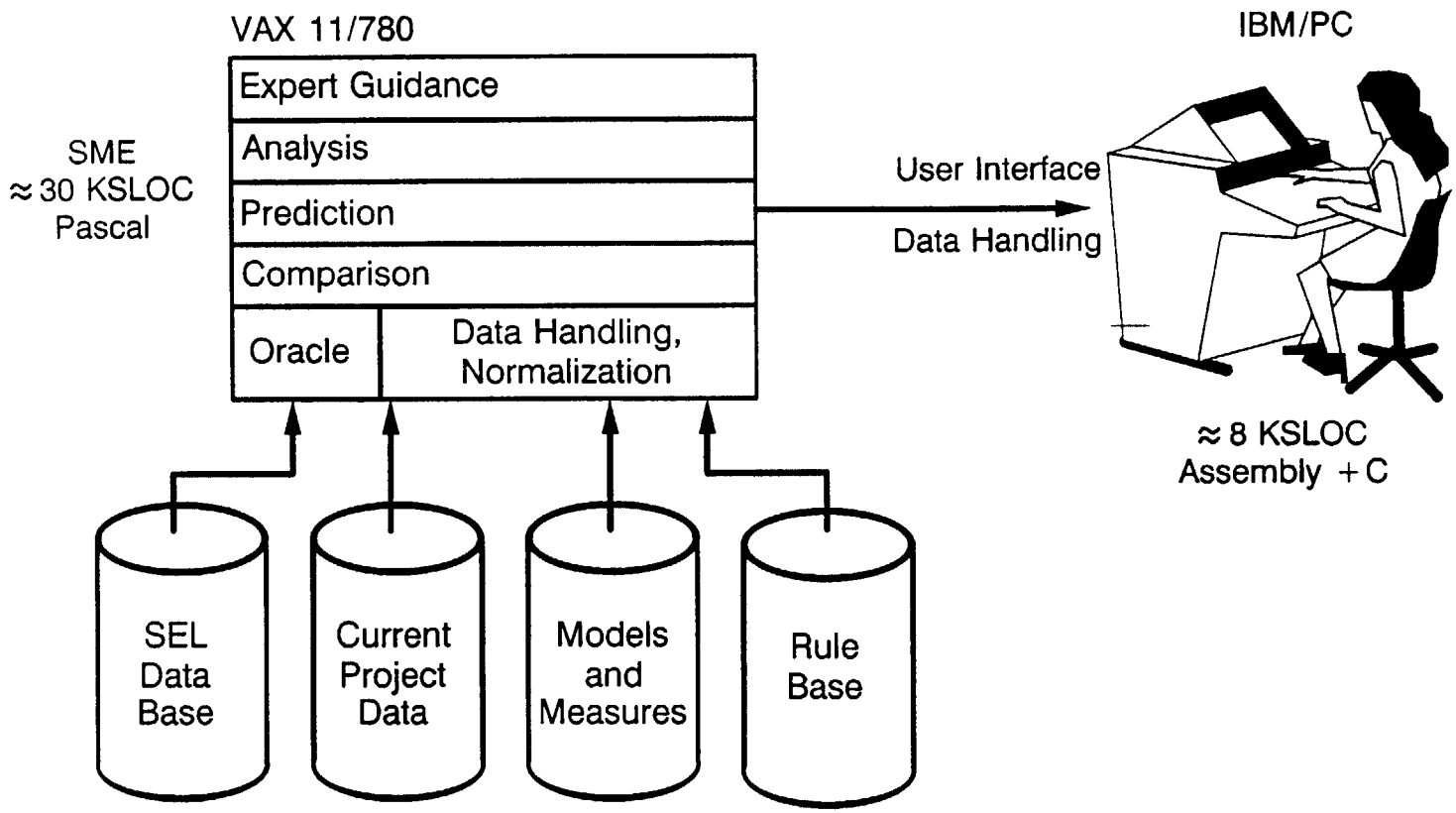
Software development rules have been developed based on previous research and the knowledge of experienced software managers

*Ramsey/Basili: 1986
Valett/Raskin: 1986
Ongoing

J. Valett
NASA/GSFC

C218.007

SME ARCHITECTURE

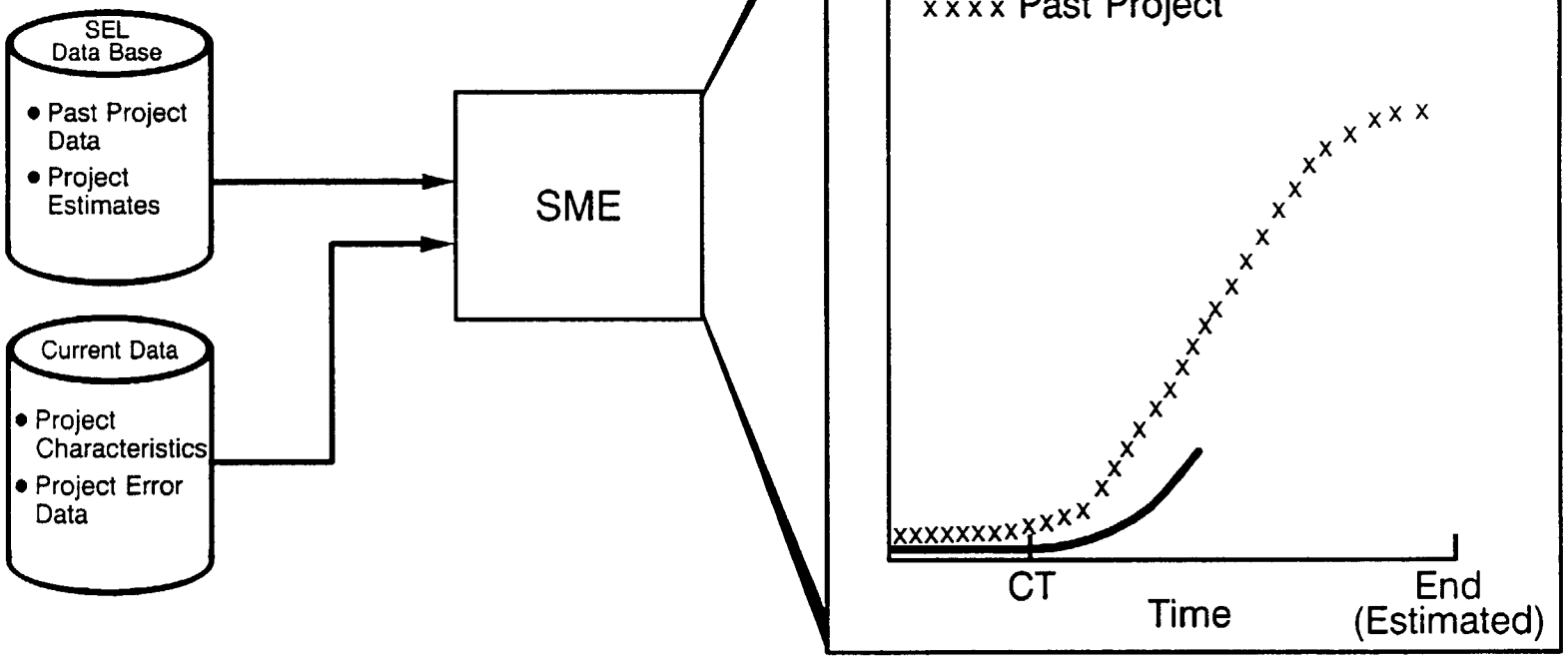


SME utilizes the SEL data base, models, and the rule base to provide managers with comparison, analysis, prediction and expert guidance

C218.008

EXAMPLE OUTPUT

How does my project **COMPARE** with one past similar project?

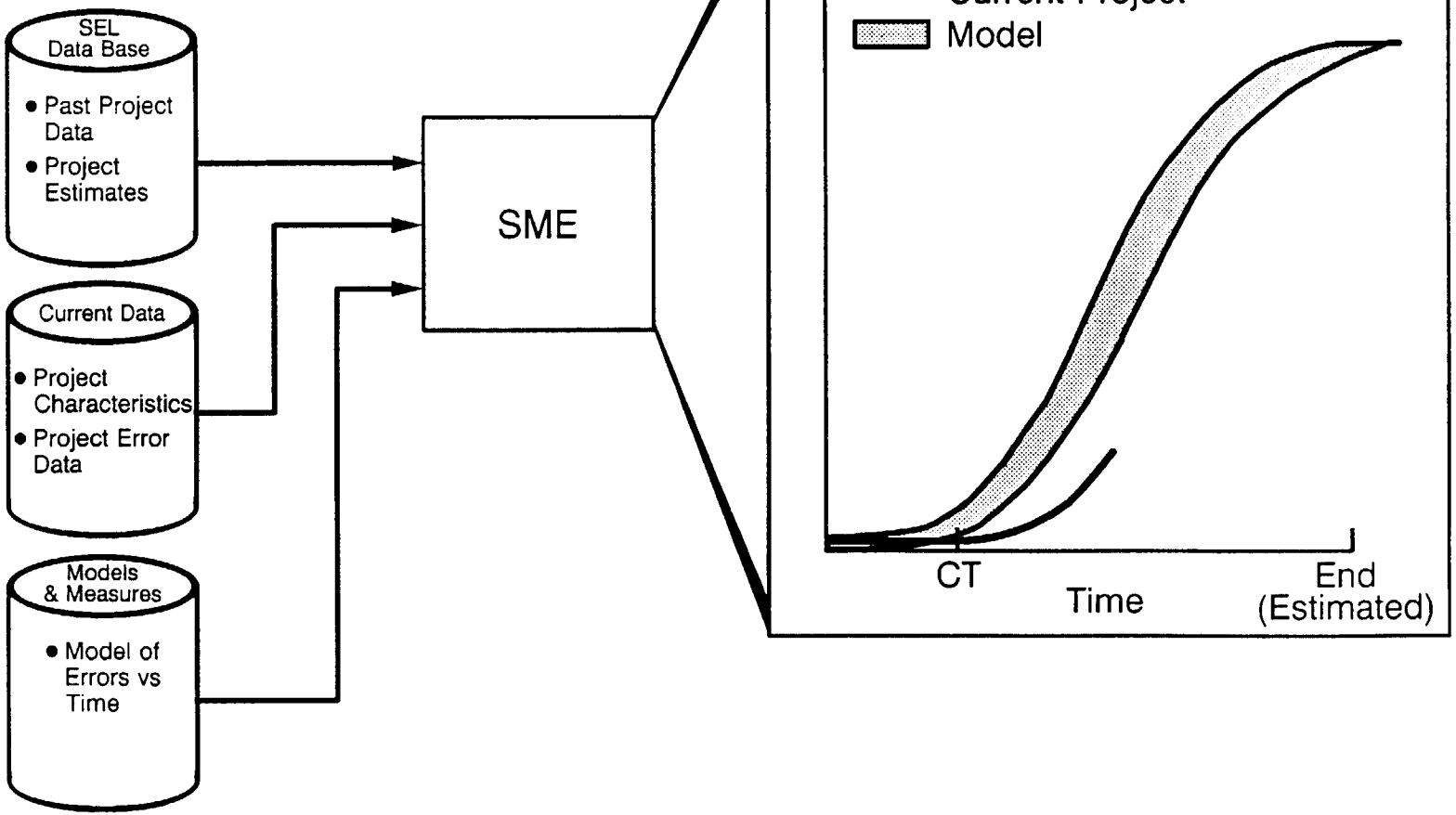


J. Valett
NASA/GSFC

C218.009

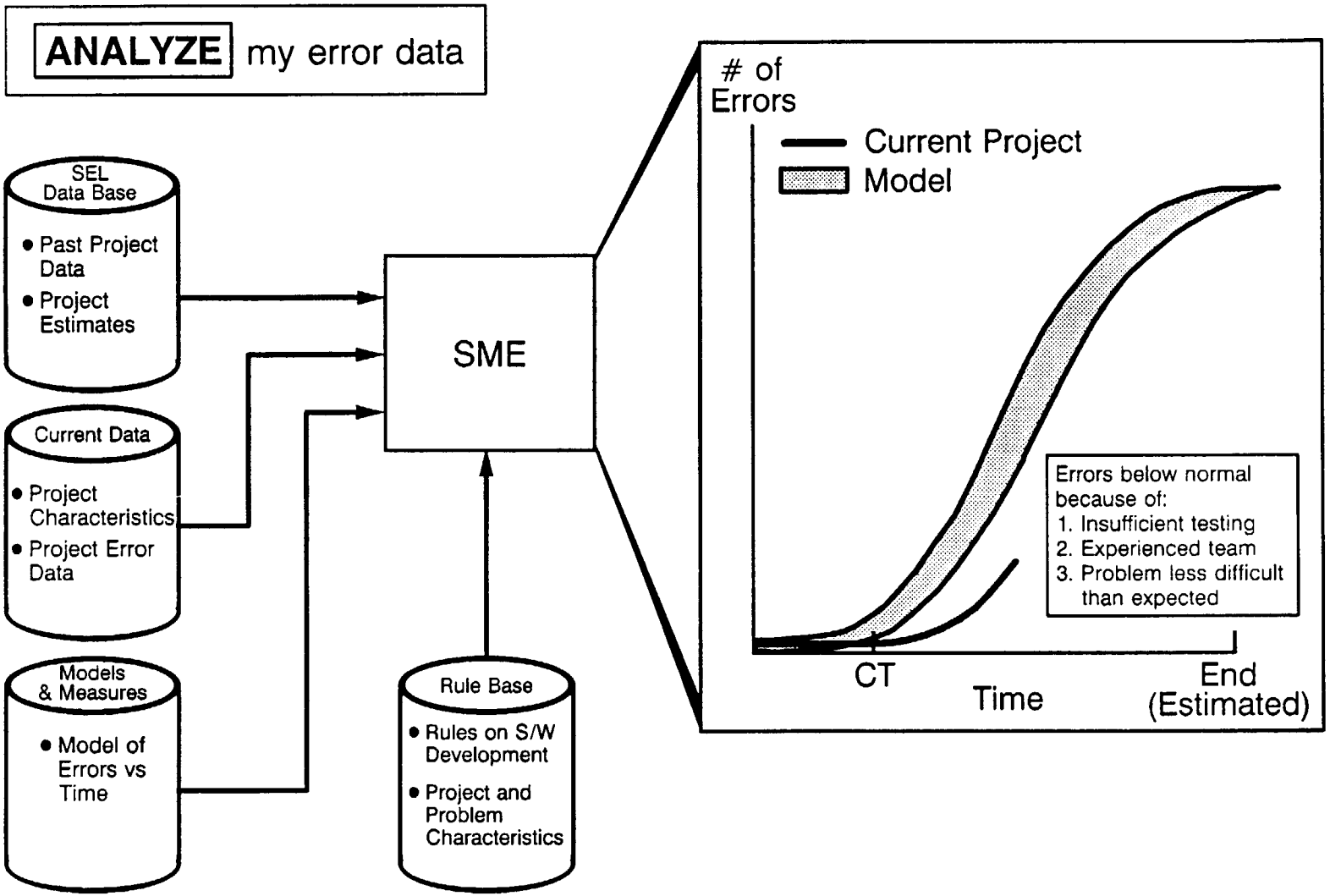
EXAMPLE OUTPUT

How does my project **COMPARE** with the typical project in the environment?



C218.010

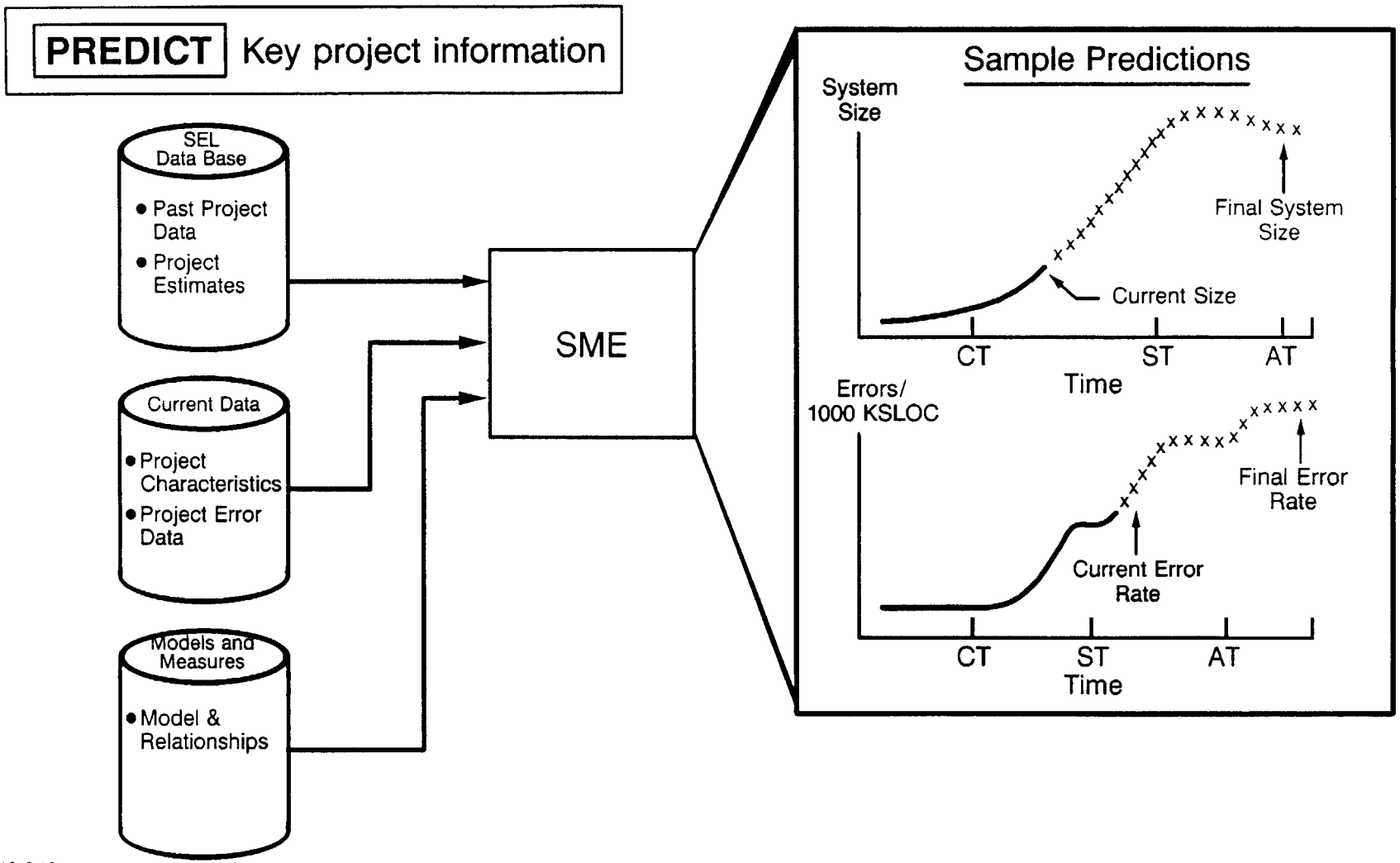
EXAMPLE OUTPUT



J. Valett
NASA/GSFC

C218.011

EXAMPLE OUTPUT



C218.012

FINAL POINTS

1. Software Measurement Data
   - Used as feedback too infrequently
   - Easy to get data can be very useful as a management aid
2. The Maturing of SME
   - SME concept is to continually evolve
   - Dynamically develop models and relationships
   - Learn "new" rules
   - Improving/refining knowledge of environment
3. Focus
   - Assumes waterfall life cycle model
   - Assumes homogeneity of problems/environment
     BUT...
     SME will evolve
4. Currently
   - Not available for general use/distribution, yet
   - Evaluation of applicability/accuracy underway

PANEL #2

SOFTWARE MODELS

R. Tausworthe, Jet Propulsion Labortary
T. Henson, IBM
W. Cheadle, Martin Marietta